

# A Scalable Mobility-Centric Architecture for Named Data Networking

Aytac Azgin, Ravishankar Ravindran, Guoqiang Wang  
Huawei Research Center, Santa Clara, CA, USA  
{aytac.azgin,ravi.ravindran,gq.wang}@huawei.com

**Abstract**—Information-centric networking (ICN) proposes to re-design the Internet by replacing its host centric design with an information centric one, by establishing communication at the naming level, with the receiver side acting as the driving force behind content delivery. Such design promises great advantages for the delivery of content to and from mobile hosts. This, however, is at the expense of increased networking overhead, specifically in the case of Named-data Networking (NDN) due to use of flooding for path recovery. In this paper, we propose a mobility centric solution to address the overhead and scalability problems in NDN by introducing a novel forwarding architecture that leverages decentralized server-assisted routing over flooding based strategies. We present an in-depth study of the proposed architecture and provide demonstrative results on its throughput and overhead performance at different levels of mobility proving its scalability and effectiveness, when compared to the current NDN based forwarding strategies.

**Index Terms**—Information-centric networks, named-data networking, mobile content delivery

## I. INTRODUCTION

*Information-centric Networking* (ICN) is a new networking paradigm that addresses the shortcomings of the current Internet architecture by shifting the focus from the host-centric communication model to a content-centric one [1]. ICN uses a unique –flat or hierarchical– naming convention to name content, which represents the main driving force for information dissemination.

Multiple architectures have so far been proposed to guide the development of ICN (see [2], [3] for a detailed overview). Architectures for information-centric networks are uniquely defined by how they handle *naming* and *name resolution*. Our research focuses on one of those proposals, namely the *Named-data Networking* (NDN) proposal, and addresses one of its major concerns, namely the *mobility*.

NDN assumes hierarchically structured names (consisting of any number and size of *components*) to support scalable routing and utilizes request/response type message (referred to as *Interest* and *Data*) processing at every hop. In NDN, each packet carries a name that can be used to identify a content, service, host, or user. Content authenticity is provided by using digital signatures that are delivered with the content. Due to its flexibility to use broadcast medium efficiently and to support loop-free forwarding on multiple interfaces, NDN is a good match to deliver (or acquire) content to (or from) mobile hosts.

However, despite the inherent advantages NDN possesses to support ad hoc networking by resolving content requests to location using online forwarding strategies, overhead associated with re-routing Interests to mobile hosts can be overwhelming, thereby, limiting NDN's efficiency for especially delivering

mobile-originated content. Thus, mobility can be considered as a major obstacle in creating a scalable network architecture based on NDN.

Mobility concerns for information-centric networks have generally been addressed by introducing location-aware routing to content delivery. However, these solutions typically assume the presence of *Global Resolution Servers* to handle the host mobility or work on the flat-name space (e.g., [4], [5]), hence, they are not directly applicable to NDN, which fundamentally does not distinguish between an entity identifier and its location. In [6] the authors briefly suggest the use of *forwarding hints* as a guideline to enable location-driven forwarding in named-data networks, which represents our starting point to develop a comprehensive mobility solution to supplement the current NDN architecture.

We can state our contributions in this paper as follows. We propose a scalable and stable mobility solution for the NDN architecture based on location-centric forwarding, by separating content names and network addresses to enable *iterative-binding*. Proposed solution uses a decentralized resolution architecture, i.e., distributed *name resolution servers*, along with in-band mobility awareness of named entity to provide dynamic name-location mappings, which in turn enables quick-recovery of the named-data path during mobile-driven content delivery. The proposed solution achieves resource efficient forwarding by avoiding *network flooding* during content delivery after handovers. The proposed solution is backward compatible as it allows the proposed extensions to be applied to mobile entities, while non-mobile entity traffic can be treated in the standard manner; this allows mobility to be treated as a *service*. The extensions also support policy-based routing by controlling intra- and inter-domain routing within the network, in the sense that the proposed data structures can be invoked only if required compared to today's situation where all mobile entities are treated alike.

To evaluate our solution, we developed extensions to the *ndnSIM* simulator [7] and analyzed the performance of the proposed solution by comparing it to the default mobility-driven NDN forwarding policies (i.e., *flooding-based policies*). We tested the proposed solution under various network topologies and mobility scenarios and observed significant improvements in the scalability performance while achieving comparable performance, in throughput, to the current NDN forwarding policies.

The rest of the paper is organized as follows. In Section II we briefly explain the NDN architecture and mobility solution for the NDN. We present the proposed forwarding architecture in Section III. We analyze the performance of our solution in Section IV. We discuss the practical considerations addressing

scalability, storage, and security concerns in Section V. Section VI concludes our paper.

## II. MOBILITY IN NAMED-DATA NETWORKS

In NDN, to support name-based routing, each node is equipped with three components: (i) *Content Store* (CS), (ii) *Pending Interest Table* (PIT), and (iii) *Forwarding Information Base* (FIB). CS is the local cache used to store content in an NDN router. Anytime a host receives an Interest for a locally cached content, a Data packet is created in response and forwarded through the incoming interface(s) for the Interest<sup>1</sup> before the Interest is discarded. PIT stores the set of active (or pending) Interests forwarded by the host and waiting for the corresponding Data to be delivered. PIT entries track the incoming faces for the received Interests and the entries are created per content, *i.e.*, any subsequent Interest for an active PIT entry is suppressed at the local host, and the current PIT entry is updated with the incoming face information. Also note that, PIT helps prevent the formation of routing loops. For that purpose, each Interest carries a random nonce to detect duplicate requests. Anytime a host receives a matching Data for a pending Interest, received Data packet is forwarded along the incoming face(s) indicated by the PIT entry, before the request is removed from the PIT. FIB aggregates forwarding information at each host, and consists of entries mapping content names to outgoing faces. To select the outgoing faces matching the prefix of an Interest, FIB uses the longest prefix matching.

To handle mobility, NDN typically relies on overhead-heavy flooding based strategies, which introduces major scalability concerns [8]. The authors in [6] address this problem by proposing the use of forwarding hints to direct requests to the content source. For that purpose, the authors aim to bring the hierarchical DNS structure to NDN, by proposing separate name and resolution servers to provide the necessary mappings (see [9] for details). However, because of the iterative/recursive lookups, the proposed approach may introduce non-negligible latency (multiple RTTs), and synchronization may be needed to support *entity* (*i.e.*, user, device, content, or service) mobility.

Our solution, on the other hand, expands the core NDN architecture and creates a mobility-based solution from within, by relying on a completely decentralized architecture, where all the operation follows the NDN principles and utilize only Interest/Data exchanges to support host mobility. The principles of our solution are listed as follows:

- *Forwarding scalability*: FIB size should be independent of the number of mobile entities. We can achieve this by requiring to keep object state only at the edges (*i.e.*, gateway points), and perform routing at the core network based on locator prefixes.
- *Control overhead*: Updates due to mobility should be local and should not affect routing/forwarding convergence. We can achieve this by using a *Local Controller* (within each domain/AS) that is capable of resolving any mobile entity using its home binding.

<sup>1</sup>NDN uses the term *face* to represent the interface over which a packet is received or delivered.

- *Intra- and Inter-session mobility*: The architecture should support both intra- and inter-session mobility (*i.e.*, changing location during a session or between sessions, where the term *session* refers to the delivery of a whole content, file, video, etc.). While *inter-session mobility* is handled through an update to/from the *Home Controller*, *intra-session mobility* is enabled by introducing a *Mobility-Update* tag, which triggers an update at all the related consumer-attached Service Routers to re-resolve the locator for the mobile entity.
- *Mobility granularity*: As producer mobility incurs cost in terms of control infrastructure, it should be realizable as a *service*, hence granularity to support mobility of any *entity* should be supported. Further differentiation can be supported in terms of geographic span of mobility, latency, etc.

In short, our solution is complimentary to NDN to handle mobility in a scalable manner. The enhancements are optional, hence, should be compatible with existing implementations as well. Next, we present our architecture.

## III. PROPOSED ARCHITECTURE

Leveraging ICN's name-based mobility, the proposed architecture allows any meaningful space of the name hierarchy to be mobile. For instance, in Figure 1, Alice's name space can be recorded as follows. Alice can choose all of her devices, `"/Alice_id"`, or one of her devices, `"/Alice_id/Alice_dev_id"`, to be mobile, or a subset of her device's content space, `"/Alice_id/Alice_dev_id/Alice_content_X"`, to be mobile. User enables this by actively registering this name space to the network, where the mobility service enabled by the provider allows the entities under the name space to be accessible anywhere on the Internet. This motivates a scalable mobility architecture. The proposed solution achieves this objective by utilizing four essential building blocks: *Local Controllers* to provide name-to-locator mappings and manage control overhead, *Fast Path Tables* (which are used by the designated routers to control information flow in the network) to address forwarding scalability, *Forwarding Labels* and *Mobility Tags* to address inter- and intra-session mobility at different mobility granularities. We observe the resulting Interest/Data packet formats in simplified view in Figure 2, which shows the additional components integrated into each packet format.

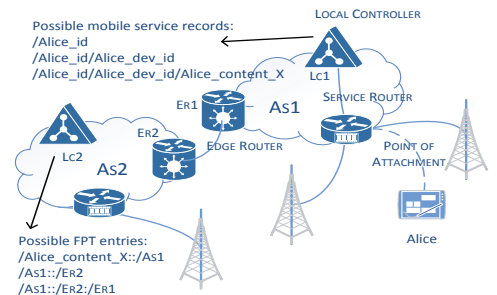


Fig. 1: ICN mobility example.

We consider a basic networking hierarchy, which consists of a given number of domains or *Autonomous Systems* (AS). Each domain/AS is assigned a *Local Controller* (LC) service carrying a domain designated prefix, *e.g.*, LC for AS-(ID:11) is assigned

INTEREST TYPE	ENTITY ID	MOBILITY FLAGS (MS-TAG & MU-TAG)	FORWARDING LABEL (RE-WRITABLE)
---------------	-----------	----------------------------------	--------------------------------

DATA	ENTITY ID	MOBILITY FLAG (MU-TAG)
------	-----------	------------------------

Fig. 2: Updated protocol data unit formats with *mobility flags* and/or *forwarding label*.

the prefix “/LOCALCONTROLLER:AS11”. LCs are responsible for mapping local and remote entity names to domain/router identifiers. If the entity is local (*intra-Domain* or *intra-AS* delivery), it resolves to *Service Router* (which can also act as a *Point of Attachment*<sup>2</sup>), whereas, if the entity is remote (*inter-Domain* or *inter-AS* delivery), it maps to local domain’s egress router identifier. Each host is statically or dynamically assigned to a designated LC (referred as *Home Controller*), which stores up-to-date information regarding its users (we will explain shortly the process to acquire such information). Also note that, proposed system allows-and supports for-the endpoints to change their Home Network bindings on-the-fly, during content delivery.

Per the NDN requirements, hierarchical names are used to identify the entities<sup>3</sup>. In the following, we pursue our discussion with respect to host mobility, which can be generalized to mobility of other named entities too. Each host is assigned a unique identifier representing the association of a user to its home network (e.g., “/AS:HOME/DEV:ID” with “Dev:Id” also including the “Host:Id”). Complete identifier, *if available*, is only needed during REGISTRATION. Network identifier, for an endpoint, is not considered *a priori* requirement to support successful location discovery. Using an identifier, however, is the preferred choice to minimize control overhead.

We assume an LC to have access to information directly related to communication taking place within its domain (e.g., home controller or locator information on content being published within the domain). Hence, no direct *synchronization* is assumed to exist among the LCs. Active domain information on visiting hosts is flushed on a regular basis, as soon as the host leaves the controller’s domain, to minimize access to outdated information. However, we allow the *Remote Controller* to store information on *Home Controllers* representing the visiting hosts for longer periods to minimize overhead associated with the *Initial Discovery Phase*.

We utilize *Forwarding Labels* to route packets in a controlled manner. Note that, routing is only needed to deliver the Interest packets, as Data packets follow the reverse path, unless tunneling is used to alter the Data path. Forwarding label is similar to content prefix in the sense that, when used, it provides sufficient information on the next physical or logical hop (i.e., gateway points). However, unlike a content prefix, forwarding label is used as a dynamic tag within the Interest that is regularly updated (at the service routers and gateway points) along the path to content source. To support the use of forwarding labels, at each service or edge router, we utilize a *Fast Path Table* (FPT) that carries

the mappings corresponding to content prefixes and forwarding addresses.

Since handling mobility with FPT and forwarding labels incurs memory and computational cost, forwarding decisions based on FPT can be limited to traffic tagged as being part of the *Mobility Service*, whereas for the other services, default routing based on FIB can be used. Specifically, because of the overhead associated with the use of forwarding labels, we consider its implementation as a *Service* (which we call as the *Forwarding Label-Service*, or the *FL-Service*) and limit its use to mobile-only scenarios. Furthermore, any mobile user that wishes to use the *FL-Service* indicates its intent during REGISTRATION by setting a single-bit *Mobility Service* tag (*MS-tag*) contained within the registration message. In addition, a consumer can also enable this tag along with the mobile entity’s unique name to invoke the *FL-Service*, which helps an NDN router differentiate between mobile and non-mobile entities. In scenarios where the *FL-Service* is not explicitly defined, forwarding is strictly based on the core NDN policies. *FL-Service*, however, is capable of taking full advantage of NDN’s strengths, i.e., in-network caching and the available FIB entries.

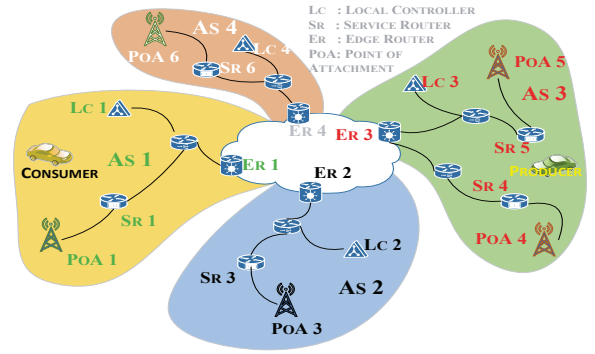


Fig. 3: A simple networking scenario with two mobile endpoints (one *Consumer* and one *Producer*) and four Autonomous Systems (ASs).

The proposed framework and the steps taken to initiate content delivery between the involved parties are explained using the scenario shown in Figure 3. For the given example, we use four Autonomous Systems with a single Consumer node, residing in AS1 and a single *Producer* node, residing in AS3. We assume that AS2 represents the Home Network for the *Producer* node. We also assume that *Producer* has already registered with its Home Network and was assigned the identifier “/AS2/PRODUCER:ID/”. Finally, we assume that *Producer* initiates the remote registration phase during its attachment process in the foreign domain as in AS3. Also, note that, we explain our solution, from the perspective of a mobile *Producer*, which is the more challenging scenario. Additional modifications needed for the mobile *Consumer* case are omitted as they share similar traits to the mobile *Producer* case, in regards to local updates, to ensure that stale information at the *Service Routers* and *Local Controllers* are promptly taken care of.

#### A. Registration Phase

- STEP I: *Registration Phase* initiates with *Producer* sending a REGISTER message for its content, referred simply as

<sup>2</sup>In the paper, we separate the *Point of Attachment* and the *Service Router* to clearly identify the different functionalities assigned to each by the proposed framework.

<sup>3</sup>Hereafter, we will use entity/content interchangeably.

/PREFIX, to its host network's LC by also including its complete identifier "/PREFIX:/AS2/PRODUCER:ID", where /AS2/PRODUCER:ID represents the home binding identifier for the entity /PREFIX. Here, PRODUCER:ID can be the device-ID, and if the device itself is mobile then /PRODUCER:ID can be considered as part of /PREFIX (hence, "/AS2" is good enough for the home binding). Furthermore, the /PREFIX itself can be globally routable or not, as it is resolved using the LC infrastructure. In short, assuming that the hosts learn the minimum control name space to interact with the POAs during mobile attachment, *Producer* sends a registration message to POA5 with the prefix "/POA5/REG"<sup>4,5</sup>. Using the FIB/FPT entries, registration message is forwarded towards LC3 through POA5 and SR5. SR5 also updates the forwarding label, by including its address to inform LC3 on the identity of the Service Router associated with the *Producer*. After LC3 receives the registration message, if the *MS-tag* is set, local database (L-DB) is updated with the entry "/PREFIX::SR5:ADDRESS::/HOME:AS2"<sup>6</sup>, where the third component specifies the Home Network for "/PREFIX". Note that, FPT or L-DB entries by default require at least two inputs, *prefix* information and *locator* information. We use double colon to separate the entries. Similarly, if the *MS-tag* is set in the registration message, FPT tables at POA5 and SR5 are also updated, e.g., by adding the entry "/PREFIX::PRODUCER" to FPT table at POA5, and the entry "/PREFIX::POA5" to FPT table at SR5, upon receiving the acknowledgement from LC3.

- STEP II: *Proactive Update Phase* initiates with LC3 sending a ROUTE-UPDATE message to the Edge Routers (ERs) (ER3, for the given scenario) to update their FPT table with the entry "/PREFIX::SR5" so that any Interest received by the ERs targeting "/PREFIX" can be immediately forwarded to SR5. Note that, at the ERs, in addition to proactive updates, we can also use reactive updates, with the ER making a request for the forwarding information. However, it requires the initial traffic to be forwarded to the LCs to minimize latency, and additional overhead to keep track of entries to avoid sending recurring route requests for non-local *Producers*.
- STEP III: *Foreign Home Registration Phase* initiates with LC3 sending LC2 a HOME-REGISTER (HREG) message for "/PREFIX", once again using the complete identifier info for *Producer*. Prefix for the HREG message is given by "/LOCALCONTROLLER:AS2/HREG". After LC2 receives the registration message, it updates its L-DB as follows:

- If an active entry is found in LC2's L-DB for *Producer*, corresponding to a different AS,

say AS4, the entry is updated as follows: "/PREFIX::REMOTE:AS3::REMOTE:AS4", where the third component indicates the previous AS the *Producer* was in. Additionally, LC2 sends a FLUSH-REGISTER message to LC4 (which will be explained shortly).

- If no active entry is found in LC2's L-DB for *Producer* or *Producer* was previously in AS2, L-DB at LC2 is updated with the following entry: "/PREFIX::REMOTE:AS3::/".

Note that, all communication in the network utilize the core Interest-Data message exchange procedures as defined by NDN. Hence, to support a reliable control message exchange, each registration/update message is followed by an acknowledgement. By jointly utilizing (content) name and (forwarding) label fields in the Interest, we can deliver all the necessary information to LCs.

## B. Content Delivery Phase

Assume that *Consumer*, who is currently connected to POA1, wants to receive "/PREFIX" published by *Producer*. Also assume that no SR or LC in AS1 has any FIB/FPT/L-DB entry for "/PREFIX".

- STEP I: *Consumer* starts *Content Delivery Phase* by sending an Interest for "/PREFIX" to POA1, who then checks, in order, its CS and PIT to find a matching entry for "/PREFIX". Since no entry is found, POA1 forwards the Interest to SR1. After receiving the Interest, SR1 performs a more detailed check including searching for FPT (and, if necessary FIB) entries. Per our assumption, since no entry is found, SR1 creates a ROUTE-REQUEST (RREQ) message with the prefix "/LOCALCONTROLLER:AS1/RREQ" and sends it to LC1. Furthermore, SR1 updates its local *request waiting list* (RWL) for the request messages it creates (similar to the PIT entries) to avoid retransmitting the same request to LC1. Any further Interests targeting "/PREFIX" are queued at SR1 until a response to the first RREQ message is received.
- STEP II: After LC1 receives the RREQ message, it searches for a matching entry within its L-DB. If an entry is found, request can be *unicast* to the LC of the home domain. If no entry is found, and the received content name does not identify the home binding (i.e., domain-based association), in a format similar to "/PREFIX:/AS2/PRODUCER:ID", then the request is forwarded to all the other LCs. For the given scenario, LC1 forwards RREQ to LC2, LC3 and LC4. We can reduce the discovery overhead by using a controlled name-based multicast, i.e., by grouping multiple requests into a single Interest and forwarding the Interest AS-by-AS until a match is found. However, in practice, we can expect such home mapping to be provided at the time of request. Also, similar to how it was with SR1, LC1 also implements a local RWL for the received RREQ messages.
- STEP III: After LC2 receives the RREQ message, a Data packet is created with the mapping "/PREFIX::/AS3" that points to the current location of the *Producer*.
- STEP IV: After LC1 receives LC2's response to its RREQ message, LC1 first updates its L-DB with the entry

<sup>4</sup>For the sake of simplicity, we added AS identifier to the control prefix. However, a single LC prefix, common to all domains, can be used to route Interests to a single LC within each domain, as these Interests will not be forwarded outside the domain.

<sup>5</sup>Depending on the implementation, information on published content is either included within the registration prefix as "/POA5/REG:</PREFIX>", or separately within the forwarding label.

<sup>6</sup>"ADDRESS" can refer to physical address space or name space. Hereafter, we drop the "ADDRESS" postfix from the host name and assume router name to represent any such address.

“/PREFIX::/AS3::/HOME:AS2”. LC1 also creates a Data packet in response to the received RREQ messages and alerts any SR indicated by the RWL entry associated with “/PREFIX”, before removing such entries. For the given scenario, LC1 responds to SR1’s RREQ message with the mapping “/PREFIX::/AS3::/ER1”, by also including information on the egress point, SR1 needs to use to reach AS3 from AS1<sup>7</sup>.

- STEP V: After SR1 receives a response to its RREQ message, SR1 updates its FPT with the following entries: “/PREFIX::/REMOTE:AS3” and “/AS3::/ER1”. SR1 then clears its RWL entry for “/PREFIX” and starts forwarding the corresponding Interests using the forwarding label “/ER1/REMOTE:AS3”.
- STEP V: After ER1 receives an Interest, it checks the forwarding label and determines AS3 as the targeted domain. We assume that the ERs have already populated their FPTs with mappings of “/REMOTE:AS::/NEXTHOP-ER”, thereby allowing ER1 to set the received Interest’s forwarding label to “/NEXTHOP-ER/REMOTE:AS3”. This process is repeated until the Interest is received by the gateway point to AS3, that is, ER3.
- STEP VI: After ER3 receives the Interest, it determines that the target for the received Interest resides within its domain (*i.e.*, AS3). ER3 uses its FPT to determine the address for the SR servicing the POA connected to *Producer* with “/PREFIX”. FPT-based lookup process is repeated at SR5 (and POA5, *if necessary*) until *Producer* receives the Interest, after which content delivery towards *Consumer* can proceed along the reverse path, using NDN’s breadcrumb approach.

### C. Intra-AS Handover Phase

Now, assume that *Producer* moves from POA5’s service area to POA4’s service area, triggering an intra-AS handover.

- STEP I: After the handover to POA4 completes, *Producer* sends a REGISTER message to LC3, as explained earlier. After LC3 receives the REGISTER message, it looks up for a matching entry for “/PREFIX” in its L-DB and notices that *Producer* is previously associated with a different SR. LC3 updates its L-DB entry with “/PREFIX::/SR4::/HOME:AS2” and sends (i) a ROUTE-UPDATE (RUPD) message to the ERs (*i.e.*, ER3 for the given scenario) to update their FPT tables with the entry “/PREFIX::/SR4”, and (ii) a FLUSH-REGISTER (FREG) message to SR5.
- STEP II: After ER3 receives the RUPD message, it updates its FPT table and forwards any matching Interest towards SR4.
- STEP III: After SR5 receives the FREG message for “/PREFIX”, it updates the local FPT with the entry “/PREFIX::/SR4” and forwards any new Interest targeting *Producer* to SR4 (note that, SR5 can also forward any undelivered Interest to SR4). SR5 also starts a timer to flush

its local entry. Additionally, SR5 forwards the FREG message to POA5, which flushes its FPT entry upon receiving.

Even though the default process for *de-registration* is to go through LCs, our framework also allows for *de-registration* through the POAs, to further improve the latency performance.

### D. Inter-AS Handover Phase

For the inter-AS handover, now assume that *Producer* moves from POA5’s servicing area to POA6’s servicing area, hence leaving AS3 to join AS4.

- STEP I: After *Producer* moves to AS4, it initiates the registration phase by sending the REGISTER message upstream towards LC4, and triggering updates along the path to (and at) LC4 (at POA6 and SR6).
- STEP II: LC4 sends a HOME-REGISTER message to LC2, while also sending a ROUTE-UPDATE message to ER4, which updates its FPT with the entry “/PREFIX::/SR6”.
- STEP III: After LC2 receives the HREG message, it determines a change-of-domain for the *Producer*, moving from AS3 to AS4, and sends LC3 a FREG message with the prefix “/LOCALCONTROLLER:AS3/FREG”. LC2 also includes information on the currently active domain (*i.e.*, AS4) for *Producer* in its message to LC3.
- STEP IV: After LC3 receives the FREG message, it creates a local FREG message that it sends to SR5. LC3 next sends a ROUTE-UPDATE-WITH-TIMEOUT message to ER3 requiring it to update its FPT entry to point to the new domain for “/PREFIX” for a forwarding-timeout period (depending on an estimate for the recovery timeframe). Anytime ER3 receives an Interest targeting “/PREFIX” with the wrong forwarding label (*i.e.*, pointing to AS3) during the FPT-timeout interval, the timeout parameter is reset to its default value (to ensure any *Consumer* not aware of domain change is informed).
  - The forwarding label for the incorrectly labeled Interest is updated with the correct label, and the Interest’s MOBILITY-UPDATE tag is set to 1, before the Interest is forwarded towards the correct domain by the ER (*i.e.*, AS4 for the given example). In the case of a failure related to previous network’s ER, hard timeouts are used at the *Consumer* side to force a location update through the LC.
  - When *Producer* receives an Interest with the *MU-tag* set, suggesting that the *Consumer*’s network is not aware of *Producer*’s change-of-domain, it sets the *MU-tag* within the Data packet as well.
  - When SR1 (or any relevant consumer serving SR) receives a Data packet with the *MU-tag* set, SR1 initiates the *Re-Discovery Phase* by requesting a forced RUPD from LC1, which then contacts LC2 to acquire the up-to-date domain information. Another alternative to the above approach is for the *Producer* side to include domain information within Data packets, in response to an Interest with a set *MU-tag*.

## IV. PERFORMANCE ANALYSIS

We implemented the proposed architecture in ndnSIM [7], by designing the modules and applications required for the

<sup>7</sup>Depending on the inter-domain routing policies, LC1 can also provide a complete path information for the Interests to go from AS1 to AS3, minimizing the processing overhead at the egress points.



processing of the received packets at designated hosts. We approximated the mobile handover by enabling/disabling wireless network interfaces based on the respective signal strength for the channel between a host and the access points reachable by the host. We assumed a handover latency of  $50ms$ <sup>8</sup>. For the point-to-point (wireline) links, we assumed a bandwidth of  $10Mbps$  and a propagation latency of  $10ms$ . We used constant-bit-rate (CBR) traffic model with a request rate of 20 packets per second. In our simulations, we considered grid-based topologies of varying sizes (i.e., 4AS scenario with 64 nodes, 9AS scenario with 137 nodes, and 16AS scenario with 236 nodes)<sup>9</sup>. We show the 4AS scenario in Figure 4. The 2nd and the 3rd scenarios, not illustrated here for brevity purposes, are an extension of the 1st scenario, with  $3 \times 3$  and  $4 \times 4$  AS formations.

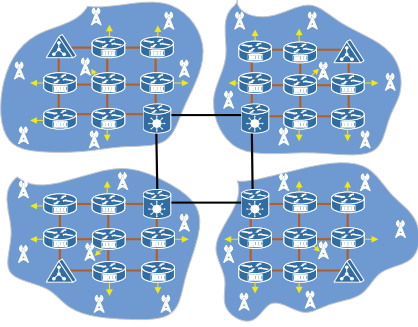


Fig. 4: “2x2 AS with 3x3 intra-AS” Grid network topology with 4AS.

We compared the performance of our forwarding solution to the *Flooding* and *Semi-flooding* (which is a modified version of *Smart-flooding* that uses Flooding in the access network to minimize the impact of timeouts) techniques (our approach utilizes the *Best-route* strategy whenever needed). In a network with unlimited resources, Flooding technique, which continuously flood the network, achieves the best performance in the end-to-end throughput, while achieving the worst overhead performance. Semi-flooding, on the other hand, uses intelligent flooding only after losses, hence represent the tradeoff between resource efficiency and end-to-end throughput.

Due to space limitation, we share our simulation results<sup>10</sup> for a single *Consumer-Producer* pair, which is sufficient to illustrate the differences between the three approaches, and specifically focused on the *Producer* mobility (i.e., *Consumer* is assumed to move at the lowest mobility level, whereas the *Producer* mobility is varied from low-to-high<sup>11</sup>). The selected mobility levels (for the

<sup>8</sup>Note that, handover latency depends on many factors, including the wireless technology utilized by the host and the type of handover initiated by the mobile host, resulting in latencies of tens-to-hundreds of ms. The chosen close-to-optimal value for the handover latency allows us to specifically focus our attention on the impact of location change.

<sup>9</sup>Grid-based topology is chosen due to its flexibility in proportionally extending the network size and measuring its impact on perceived performance.

<sup>10</sup>We ran 10 simulations using different random seeds, each of which lasts for 30 minutes, and present the average of their results.

<sup>11</sup>*Consumer*’s mobility region is limited to the top row of ASs, whereas the remaining ASs are used to represent the *Producer*’s mobility region. To approximate the worst case conditions for the proposed architecture and prevent easy access to *Producer*’s current location, one of the top row ASs—not used by the *Consumer*—is chosen as the *Producer*’s Home Network.

*Random Waypoint Model* in ns3) and their impact on handover frequencies are illustrated in Figure 5 (single-number scenario refers to constant speed mobility, whereas two-number scenario refers to mobility based on (min,max) speeds). We observe from Figure 5 that the chosen mobility levels can trigger highly unstable conditions due to frequent handovers, hence, are extremely useful to demonstrate how the three approaches perform under such conditions.

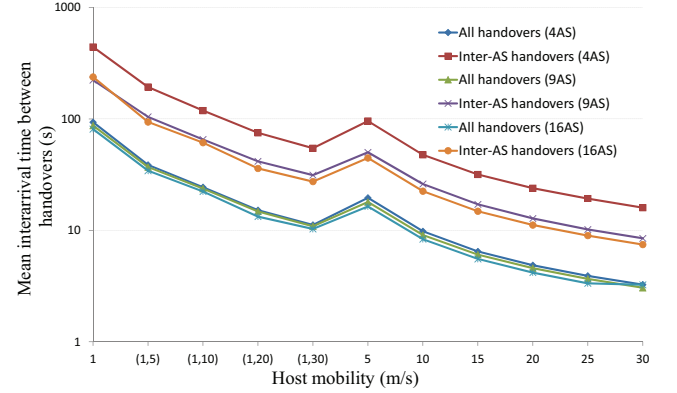


Fig. 5: Interarrival times for the mobile handovers for the three considered network topologies, also including the results for the inter-AS handovers ( $\approx 20\text{--}34\%$  of the handovers are inter-AS handovers).

#### A. Session Throughput

The CBR model used in ndnSIM assumes an always constant request rate, i.e., no traffic is rushed (including the retransmissions). Hence, throughput performance is inversely proportional to both the recovery latency after handovers (the lower the latency, the higher the throughput) and the retransmitted Interests (the higher the retransmission rate, the lower the throughput). In short, from the perspective of the delay-tolerant traffic, effective throughput represents the percentage ratio of Data packets successfully received by the *Consumer* at the end of a simulation run (i.e., the ratio of the number of Data packets received to the number of Interest packets transmitted by the *Consumer* application).

We illustrate the results for the effective throughput performance in Figure 6 (where we use the term *FastForwarding* to refer to our solution). We observe that the proposed framework achieves better than 80% throughput for all the considered scenarios, and performs significantly better than Semi-flooding, regardless of network size. Our analysis suggest that the performance of the proposed architecture can be further improved (to, for instance, better than 90% at the highest user speed for the 9AS scenario) by avoiding retransmissions experienced due to stale PIT entries triggering different paths (that may not reach the *Producer* in time). There are multiple ways to achieve that without introducing significant overhead to the system, for instance, by limiting PIT entries for the FL-based mobile traffic.

#### B. Recovery Overhead

One of our objectives in designing a location-centric forwarding technique is to minimize the overhead introduced during path

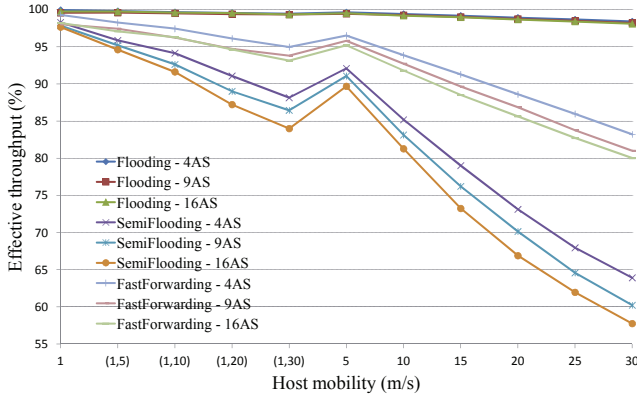


Fig. 6: Throughput performance of the proposed architecture with respect to *Flooding* and *Semi-flooding*.

recovery after mobile handovers. We can clearly see the potential impact of mobile handovers, if we examine the Interest rates shown in Figure 7. We observe significantly better results with our solution that is 5–12 times better, when compared to Flooding, and 2–5 times better, when compared to Semi-flooding.

Another way to look at the overhead performance is to measure the additional number of Interest packet transmissions used to achieve the perceived throughput performance. We show the results in Figure 8. From such perspective, we observe up to 50–200 times better performances when compared to the Flooding technique, and 5–20 times better performances when compared to the Semi-flooding technique, thereby, further illustrating the efficiency of our solution. Also note that, the increase in overhead for our solution is mostly caused by the wasted retransmission attempts leading to lower throughput, as explained earlier. By minimizing the number of such attempts lowers the percentile overhead by 60%, with much slower-less than half-increase rate between lowest-to-highest mobility levels.

In short, our approach proves to be a much more scalable solution, in both the network size and the host mobility.

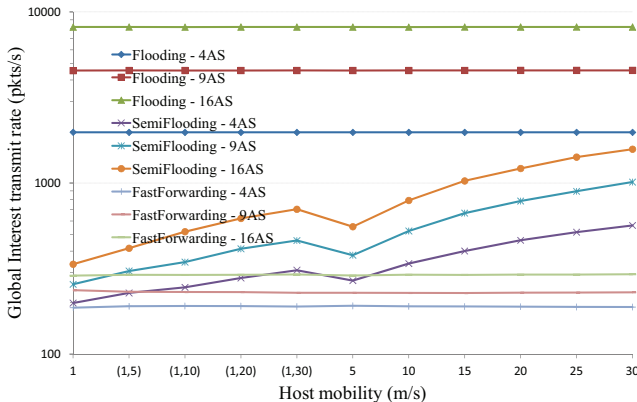


Fig. 7: Comparative results for the overall network-wide Interest rate.

Analytically, we can approximate the worst case Interest overhead (corresponding to inter-AS handover) for the proposed Mobility Service framework during the lifetime of a session as  $O(h \log N)$  (where  $h$  represents the overall handover rate, and  $N$  represents the network size), which can be determined as follows.

During an inter-AS handover, the following events take place with our approach: (i) *Producer P* registers its prefix to the new remote-domain  $D_C$ , (ii)  $D_C$ 's LC updates  $P$ 's HOME-LC (HLC), and (iii)  $P$ 's HLC updates the previous remote-domain  $P$  was registered to,  $D_P$ , which then updates its local ERs and SRs. The overhead for the first event is given by  $O(\delta_{nL})$ , for the second event is given by  $O(\delta_{nG})$ , and for the third event is given by  $O(\delta_{nL} + \delta_{nG})$ , where  $\delta_{nL}$  represents the average distance between two nodes within a local domain, and  $\delta_{nG}$  represents the average distance between two nodes within two separate domains. We can approximate the average distance between any two nodes in a network of size  $N$  using  $\log N$ , allowing us to approximate the overhead during a handover using  $O(\log N)$ . On the other hand, for the default NDN forwarding policies (specifically *Smart-flooding* based policies), the overhead during a handover is approximated as  $O(N)$ , since the handovers trigger *flooding* in the network.

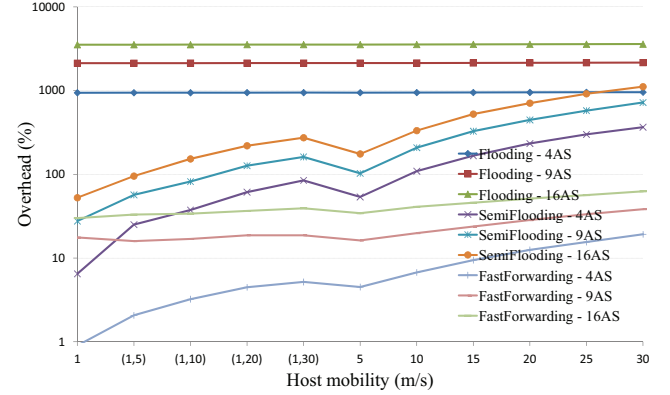


Fig. 8: Comparative results for the overhead performance (*i.e.*, percentile ratio for the additional Interest packets transmitted to support the perceived Data delivery rates).

For instance, for the synthetic topologies we considered, going from 4AS to 9AS (or 16AS),  $\delta_{nG}$  increases by 1.11 (or 1.22). Similarly, going from 9AS to 16AS,  $\delta_{nG}$  increases by 1.1. To represent the change in  $N$ , we can simply use the ratio of the number of ASs (resulting in 2.25, 4, and 1.78). Our results suggest that, change in overhead for the given scenarios are given by: for *Flooding* (2.3, 4.1, 1.8); for *Semi-flooding* (1.4–1.8, 2.2–2.8, 1.3–1.6); and, for our solution, (1.2–1.4, 1.5–1.7, 1.2–1.3). In short, *flooding*-based techniques observe an increase in overhead closer to change in  $N$ , whereas our solution observes an increase in overhead closer to  $\delta_{nG}$ , further proving its scalability.

## V. DISCUSSIONS

### A. Storage Considerations

FIB represents both the strengths (*i.e.*, flexible operation and on-the-fly routing decisions) and the weaknesses (*i.e.*, overhead) of NDN, with greater perceived impact as the network size and content availability increase. Specifically, ad hoc operation allows for greater flexibility during routing, whereas, increased database size (with variable prefix names and multiple entries per prefix) degrades the operational efficiency at each NDN capable router by increasing the perceived processing latency (see [10] for a detailed discussion on challenges in content-centric forwarding).

FPT addresses these drawbacks by partitioning the information required for end-to-end routing at different sections in the network. The major portion of the -mostly local- FIB entries, *i.e.*, prefix-to-address mappings, are stored at the L-DBs. However, due to domain-based partitioning, the number of entries stored at a given L-DB is expected to be much smaller than what is expected to be stored in the FIB of an NDN router. SRs are only responsible for keeping entries of the active hosts they serve, rather than keeping entries for the hosts being serviced by the other routers. ERs provide the backbone dependent mappings and their perceived overhead is limited in the maximum of the number of domains/Ass and the number of locally hosted *Producers*. The intermediate routers, between SRs and ERs, are only responsible for carrying the next hop-to-ER mappings, hence not anymore observing overhead proportional to the number of hosts being serviced along the downstream channel. As a result, we expect noticeable improvement in the processing latency within the network to route packets between endpoints. Specifically, by forwarding Interests on the LC-managed address-space, rather than the highly variable prefix-space, lookup latency on a typical NDN router does not anymore depend on the prefix length.

### B. Scalability Considerations

For the proposed architecture, another important concern is the performance of the LC, which is expected to service the requests of the hosts associated with the LC's domain. An LC carries prefix-to-domain mappings for the hosted *Consumers* and remote *Producers* and prefix-to-router mappings for the hosted *Producers*. Note that prefix-to-domain mappings are updated at a much lower rate (*inter-AS* handover rate) than the rate associated with prefix-to-router mappings, which change at the *intra-AS* handover rate. We can therefore approximate the request rate  $\rho$  as follows:

$$\rho = h_{AS} \times (|mC| + \kappa|RmP|) + h_{AP} \times |mP| \quad (1)$$

where  $\{h_{AS}, h_{AP}\}$  represent the mean *inter-AS* and *intra-AS* handover rate (with  $h = h_{AS} + h_{AP}$ ),  $\kappa$  represents the average number of unique *Consumer* domains requesting content from the same *Producer*,  $\{mC, mP\}$  represents the sets of mobile *Consumers/Producers* hosted by the current domain,  $\{RmP\}$  represents the set of remotely located mobile *Producers* using the current domain as a *Home* network, and  $|\cdot|$  represents the size operator. As the network size increases, we expect  $\kappa$  to converge to a small constant and we can assume  $|mC| = |mP| = |RmP| = n_m$ . Therefore, we can approximate  $\rho$  as  $h \times (1 + \gamma\kappa) \times n_m$ , where  $\gamma = h_{AS}/h$ . If we assume frequent handovers with mean inter-handover latency of 10s, and with  $\kappa \approx 1/\gamma$  and  $n_M = 1M$ , we expect a value of  $\rho = 200K$  requests per second, which is easily manageable with the current server architectures.

Also note that, because of the overall latency incurred during handovers (caused by handover and end-to-end propagation delays), we consider the requirements on processing latency to be less strict than that of NDN forwarding, thereby allowing the LC to be more flexible in its lookup operations. Furthermore, if necessary, multiple LCs can be assigned to a single domain to support load balancing within that domain. In that case, a simple

hash function that assigns prefixes to an LC can be installed on each designated router to support such features. Our framework is flexible enough to manage such scenarios with little increase in complexity.

### C. Security Considerations

There are various ways an attacker can exploit the possible vulnerabilities in our architecture by targeting the LCs. For instance, by registering non-existing prefixes to the LC as fake *Producers*, and by requesting non-existing prefixes from the LC as fake *Consumers*, attackers can overload the controllers and limit access to the legitimate requests. We next explain possible approaches we can use in such scenarios to minimize the impact of flooding attacks on the overall performance.

1) *Producer Flooding*: Our architecture assumes a *Producer* to include certain information within the registration message to identify the *Producer's* home network and authenticate the registered content. Hence, we can limit the scope of fake-*Producer* attacks through authentication failure messages received from the home networks. After an authentication failure message is received by the host network's LC, information on the fake-*Producer* can be shared with the host network's SRs, to prevent or reduce access to the matching user's registration requests.

2) *Consumer Flooding*: To prevent an attacker from hijacking the network by sending requests for non-existent prefixes, multiple approaches are possible. First, we can employ a threshold-based admission policy at the first point of entry for the incoming requests and limit the number of outstanding requests that await for the path update from the LC. Our architecture already does this to some extent, by suppressing requests targeting the same entity (*i.e.*, *Producer*) at the SRs. Second, we can use an adaptive decision policy to enforce stricter threshold values at certain SRs depending on the experienced overhead at the LCs. Since the forwarding label in a request message includes information on the entry points, LCs can aggregate the necessary statistics to quickly determine the problematic areas, and restrict access whenever needed. Third, by sending feedbacks to SRs on problematic requests, attackers can be identified in a timely manner and the information on them can be shared with other SRs within the same domain to limit the effectiveness of future attacks.

## VI. CONCLUSION

In this paper, we proposed a decentralized mobility-centric solution for Named-data Networking (NDN) to address the scalability problems that arise during the delivery of mobile content to requesting *Consumers*. The proposed solution relies on four essential building blocks to support location-driven forwarding: *Local Controller* (resolution server to provide name-to-locator mappings), *Forwarding Label* (dynamic path information inserted within the Interest to support *iterative-binding* by intelligently guiding the request towards the content source), *Mobility Tags* (single-bit flags inserted within Interest or Data packets to indicate entity mobility or mobility service) and *Fast Path Table* (database utilized at the designated routers to store name-to-locator mappings). We presented an in-depth analysis of the proposed architecture and explained in detail all the necessary steps required to initiate and maintain connectivity between



mobile end points. We implemented our solution in ndnSIM and demonstrated significant performance improvements in network scalability while achieving comparable results to *flooding* in effective throughput. We also addressed the practical considerations in regards to storage requirements, controller scalability, and security concerns, and discussed the efficiency and effectiveness of the proposed architecture.

## REFERENCES

- [1] G. Xylomenos, C. Ververidis, V. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. Katsaros, and G. Polyzos, "A survey of information-centric networking research," *IEEE Communications Surveys Tutorials*, vol. PP, no. 99, pp. 1–26, 2013.
- [2] M. F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and B. Mathieu, "A survey of naming and routing in information-centric networks," *IEEE Communications Magazine*, pp. 44–53, Dec 2012.
- [3] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, 2012.
- [4] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *ACM SIGCOMM*, 2007, pp. 181–192.
- [5] C. Dannewitz, J. Golic, B. Ohlman, and B. Ahlgren, "Secure naming for a network of information," in *IEEE INFOCOM Computer Communications Workshops*, 2010, pp. 1–6.
- [6] Z. Zhu, A. Afanasyev, and L. Zhang, "A new perspective on mobility support," NDN, Technical Report NDN-0013, July 2013. [Online]. Available: <http://named-data.net/techreports.html>
- [7] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," NDN, Technical Report NDN-0005, Oct 2012.
- [8] A. Azgin, R. Ravindran, and G. Wang, "Mobility study for named data networking in wireless access networks," in *IEEE International Conference on Communications (ICC)*, 2014.
- [9] A. Afanasyev, "Addressing operational challenges in Named Data Networking through NDNS distributed database," Ph.D. dissertation, UCLA, September 2013.
- [10] H. Yuan, T. Song, and P. Crowley, "Scalable NDN forwarding: Concepts, issues and principles," in *IEEE International Conference on Computer Communications and Networks (ICCCN)*, 2012.